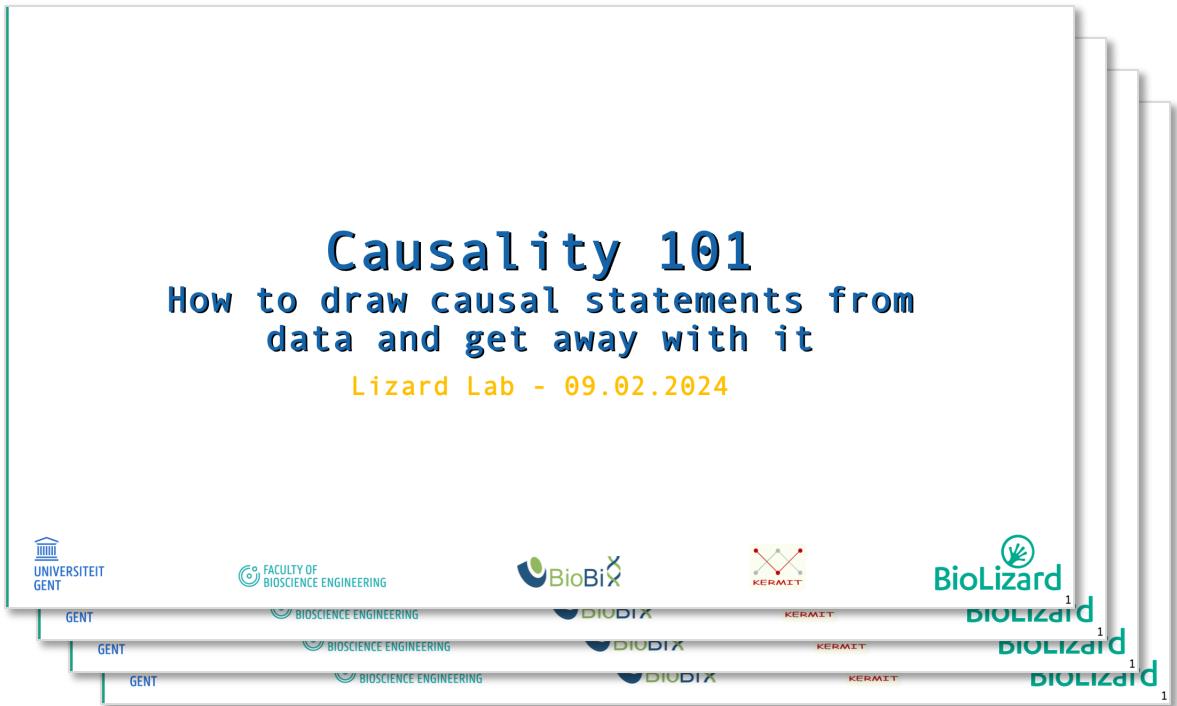


# Causality 101

## How to draw causal statements from data and get away with it

Lizard Lab - 09.02.2024

# Structure of the session



Part 1 : Why do we need causality?  
"Correlation is not causation"

It is chanted by every statistician and their mothers, but why? Let's do some experiments.

Experiment 0: Directionality

Let's say we have data on two variables, the expression of a certain gene and a disease status, and we want to see if they are correlated. Let's simulate some data assuming as ground truth that the gene is causal for the disease.

```
# simulate some data where expression causes disease
n = 100
expression = np.random.normal(0.5, 0.25, n)
disease = []

for i in range(n):
    ... # the causal part
    ... # low noise
    ... # high noise
```

CO

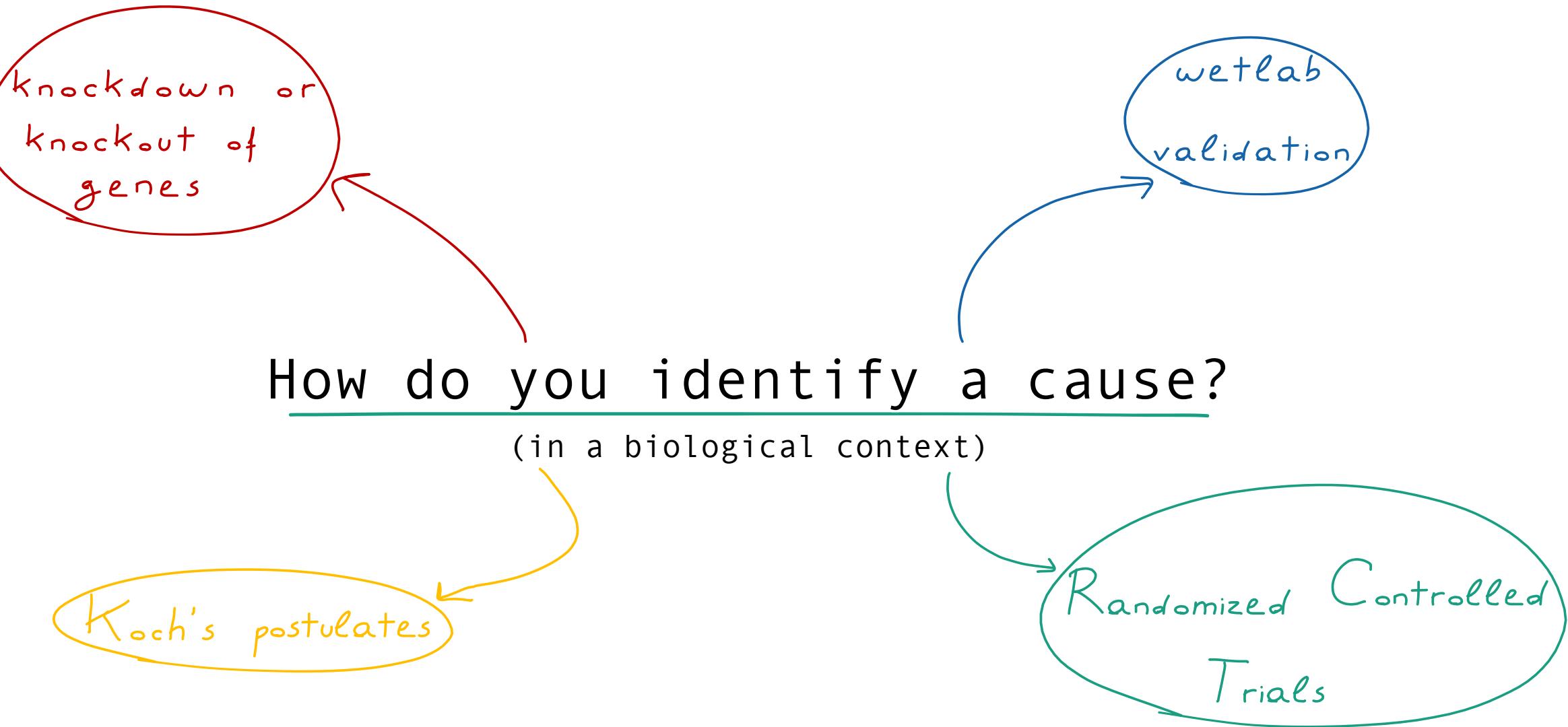
GITHUB



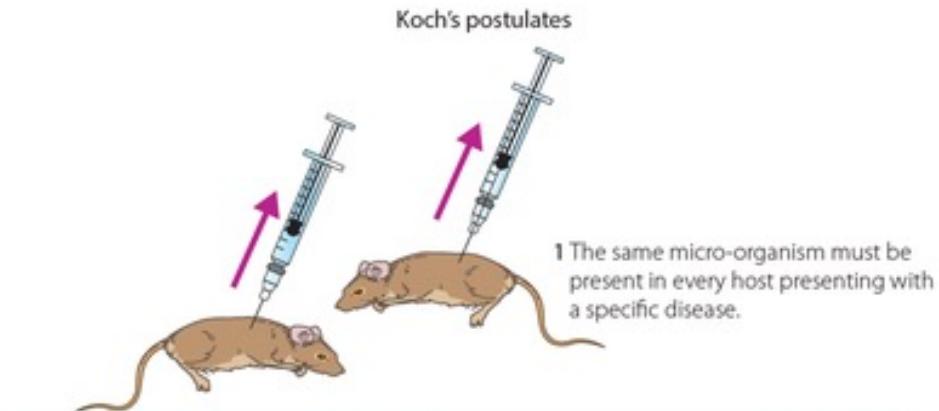


## How do you identify a cause?

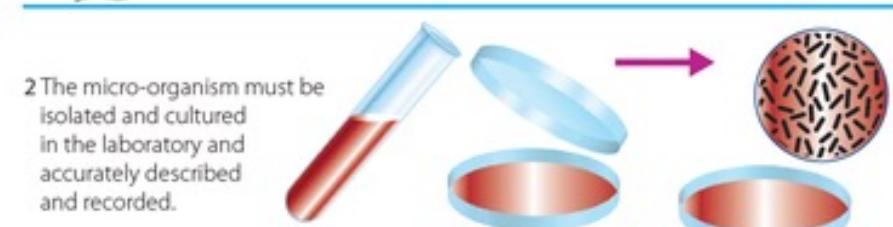
(in a biological context)



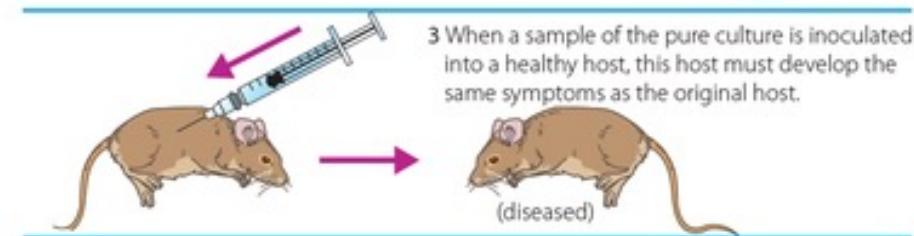
**Common**



**Culture**



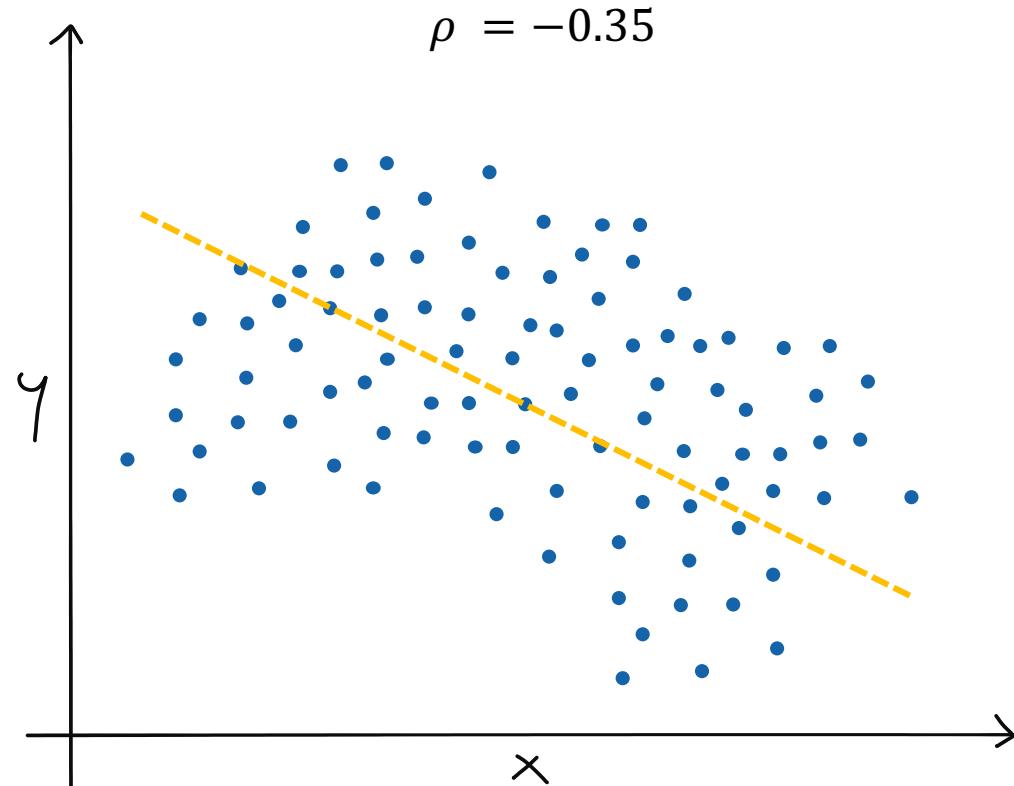
**Cause**



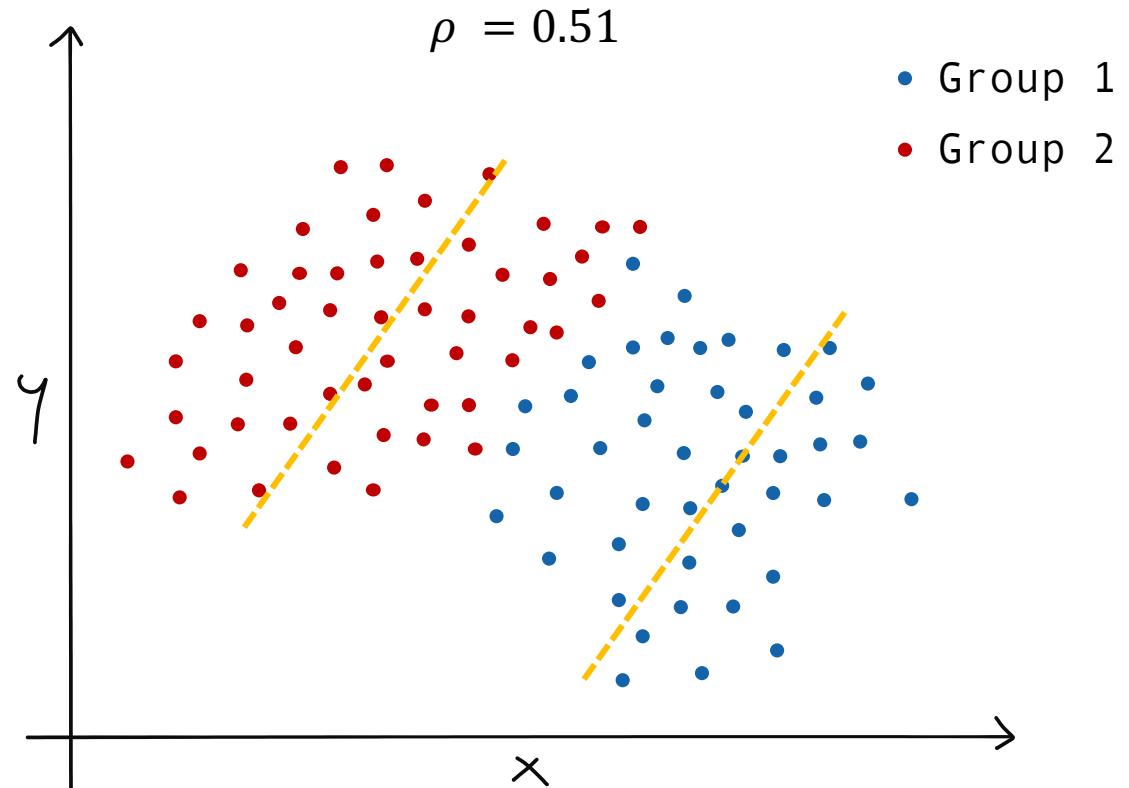
**Compare**



# Why correlation $\neq$ causation



# Why correlation $\neq$ causation



# Why correlation ≠ causation



why.ipynb

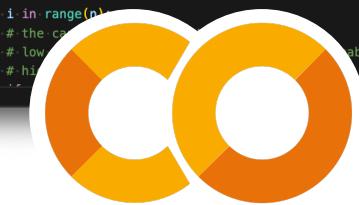
The screenshot shows a Jupyter Notebook interface with the following details:

- File Structure:** The left sidebar shows a project structure with files like 'why.ipynb', 'how.ipynb', 'CAUSALITY\_101', 'DAGs.pptx', 'environment.yml', 'README.md', and 'why.ipynb'.
- Code Cell:** The main area contains Python code imports:

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from scipy.stats import spearmanr, pearsonr, kendalltau
import seaborn as sb
from tqdm import tqdm
```
- Section Headers:** The notebook includes sections titled "Part 1: Why do we need causality?" and "Correlation is not causation".
- Text Content:** A paragraph explains the need for causality: "It is chanted by every statistician and their mothers, but why? Let's do some experiments."
- Experiment Description:** A section titled "Experiment 0: Directionality" describes simulating data where gene expression causes disease.
- Code Example:** A snippet of Python code for generating simulated data is shown:

```
# simulate some data where expression causes disease
n = 100
expression = np.random.normal(0.5, 0.25, n)
disease = []

for i in range(n):
    ... # the causal path
    ... # low noise
    ... # high noise
```



# Why correlation ≠ causation

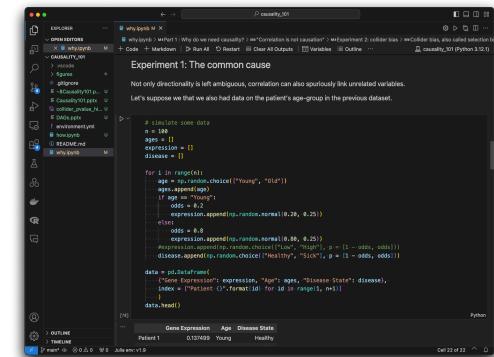


Part 1: Why do we need causality?  
"Correlation is not causation"  
It is chanted by every statistician and their mothers, but why? Let's do some experiments.  
Experiment 0: Directionality  
Let's say we have data on two variables, the expression of a certain gene and a disease status, and we want to see if they are correlated. Let's simulate some data assuming no ground truth that the gene is causal for the disease.

```
# simulate some data where expression causes disease
# ID
expression = np.random.normal(0.5, 0.25, n)
disease = []
for i in range(n):
    # the disease is independent
    # expression causes disease with 20% probability
    # high expression -> disease
    if expression[i] > 0.75:
        disease.append(1)
    else:
        disease.append(0)
```

# Why correlation ≠ causation

★ directionality

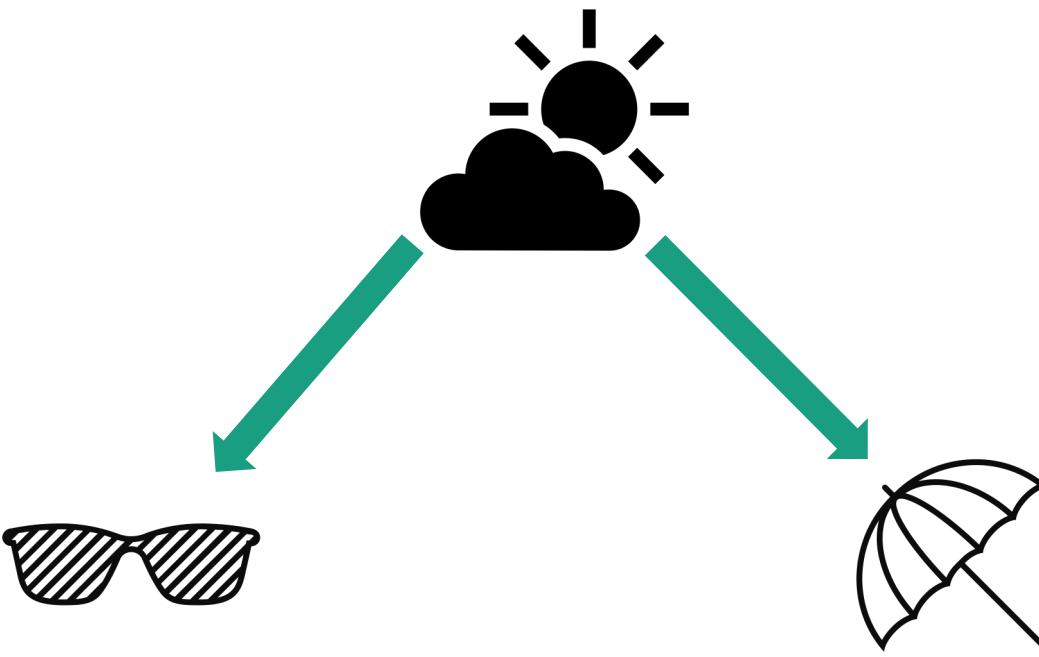


```
for l in range(2):
    if l == 0:
        age_group = choice(["Young", "Old"])
        age.append(age_group)
        if age_group == "Young":
            odds.append(0.2)
            expression.append(np.random.normal(0.28, 0.25))
        else:
            odds.append(0.8)
            expression.append(np.random.normal(0.88, 0.25))
    else:
        age_group = choice(["Young", "Old"])
        age.append(age_group)
        if age_group == "Young":
            odds.append(0.2)
            expression.append(np.random.choice([-0.05, -0.02], p = [1 - odds[0], odds[0]]))
        else:
            odds.append(0.8)
            expression.append(np.random.choice([0.05, 0.02], p = [1 - odds[1], odds[1]]))

data = pd.DataFrame({
    "Gene Expression": expression,
    "Age": age,
    "Disease State": disease
})
```

# Why correlation ≠ causation

★ directionality



```
for l in range(5):
    age = np.random.choice(["Young", "Old"])
    gene_exp.append(age)
    if age == "Young":
        odds = 8.2
    else:
        odds = 0.8
    expression.append(np.random.normal(0.28, 0.25))
    disease.append(np.random.choice(["Healthy", "Sick"], p = [1 - odds, odds]))
    disease.append(np.random.choice(["Healthy", "Sick"], p = [1 - odds, odds]))
```

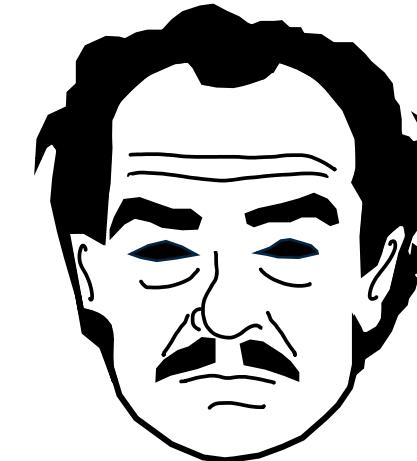
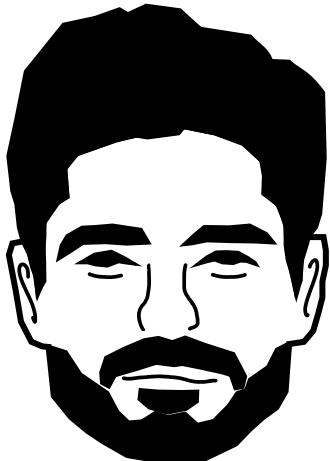
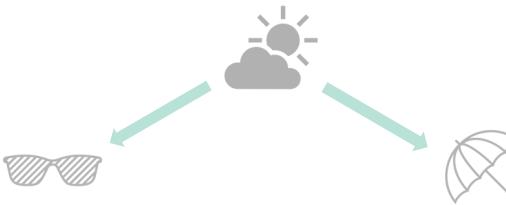
Gene Expression	Age	Disease State	
Patient 1	0.37469	Young	Healthy

# Why correlation ≠ causation

★ directionality



★ common causes



```
# we will just simulate a binary flip, where 1 = heads and 0 = tails
for i in range(10):
    coin_1 = np.random.choice([0, 1])
    coin_2 = np.random.choice([0, 1])
    if coin_1 or coin_2:
        results_c1.append(coin_1)
        results_c2.append(coin_2)

# we can print the results
print(results_c1)
print(results_c2)
```

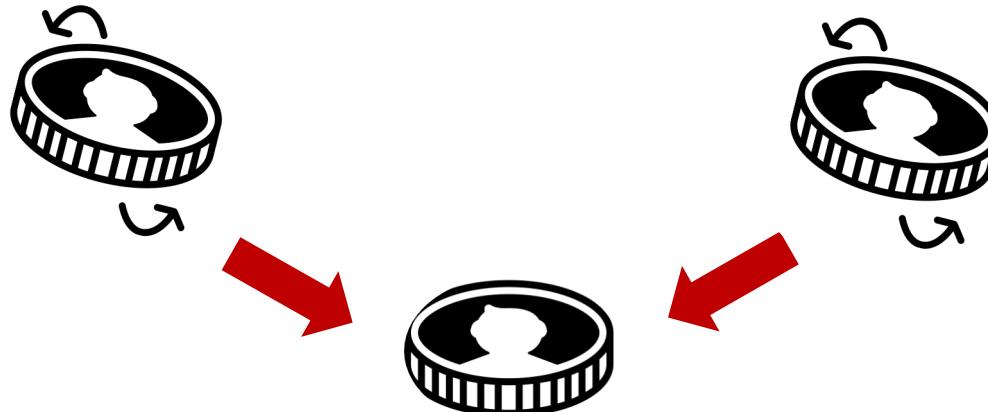
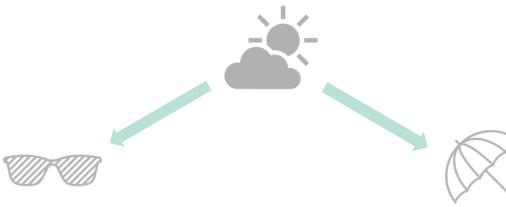
coin 1	coin 2
0	0
1	0
0	1
1	1
0	0
1	0
0	1
1	1
0	0

# Why correlation ≠ causation

★ directionality



★ common causes



```
# we will just simulate a binary flip, where 1 = heads and 0 = tails
for i in range(10):
    coin1_i = np.random.choice([0, 1])
    coin2_i = np.random.choice([0, 1])
    if coin1_i or coin2_i:
        results_i.append(coin1_i)
        results_i.append(coin2_i)
    else:
        results_i.append(0)

# we can now look at the results
results = pd.DataFrame(results, columns=['coin1', 'coin2'])
results['heads'] = np.where(results['coin1'] + results['coin2'] > 0, 1, 0)
```

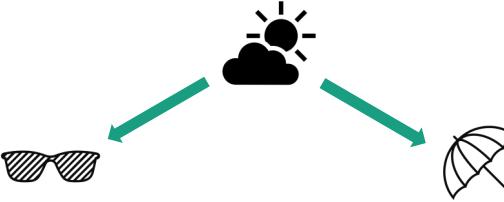
coin1	coin2	heads
1	0	1
0	1	1
1	1	1
0	0	0
1	0	1
0	1	1
1	1	1
0	0	0
1	0	1
0	1	1

# Why correlation ≠ causation

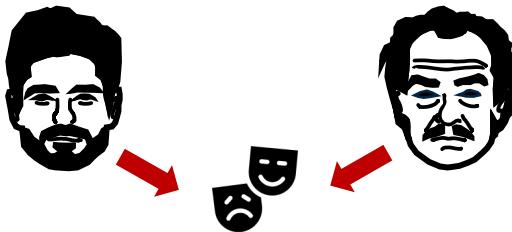
✿ directionality



✿ common causes



✿ collider/selection bias



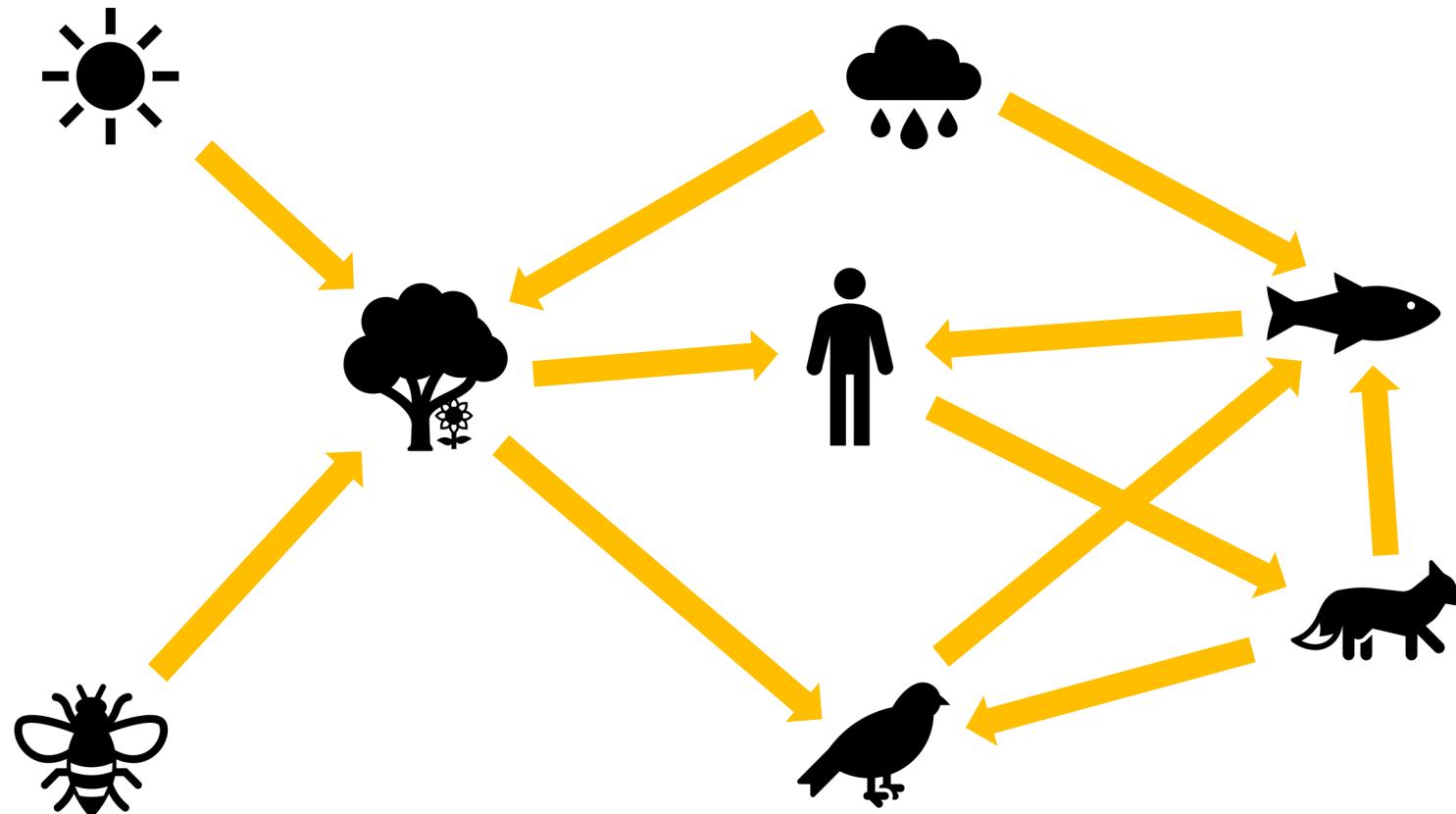
A screenshot of a Jupyter Notebook titled "collider.ipynb". The code simulates two coin flips and records the outcomes. The results show that while the common cause (coin flip) is roughly 50% heads/tails, the collider variable (heads) is approximately 75% heads, demonstrating collider bias.

```
# we will just simulate a binary flip, where 1 = heads and 0 = tails
for i in range(10):
    coin_1 = np.random.choice([0, 1])
    coin_2 = np.random.choice([0, 1])
    if coin_1 or coin_2:
        results_collider.append(1)
    else:
        results_collider.append(0)

# we can print the results
print(results_collider)
# coin 1: results_collider, coin 2: results_collider
# coin 1: [1, 0, 1, 1, 0, 1, 1, 0, 1, 1] for i in range(1, len(results_collider)+1):
# heads
```

coin 1	coin 2
0	0
0	1
1	0
1	1
0	0
0	1
1	0
1	1
0	0
0	1

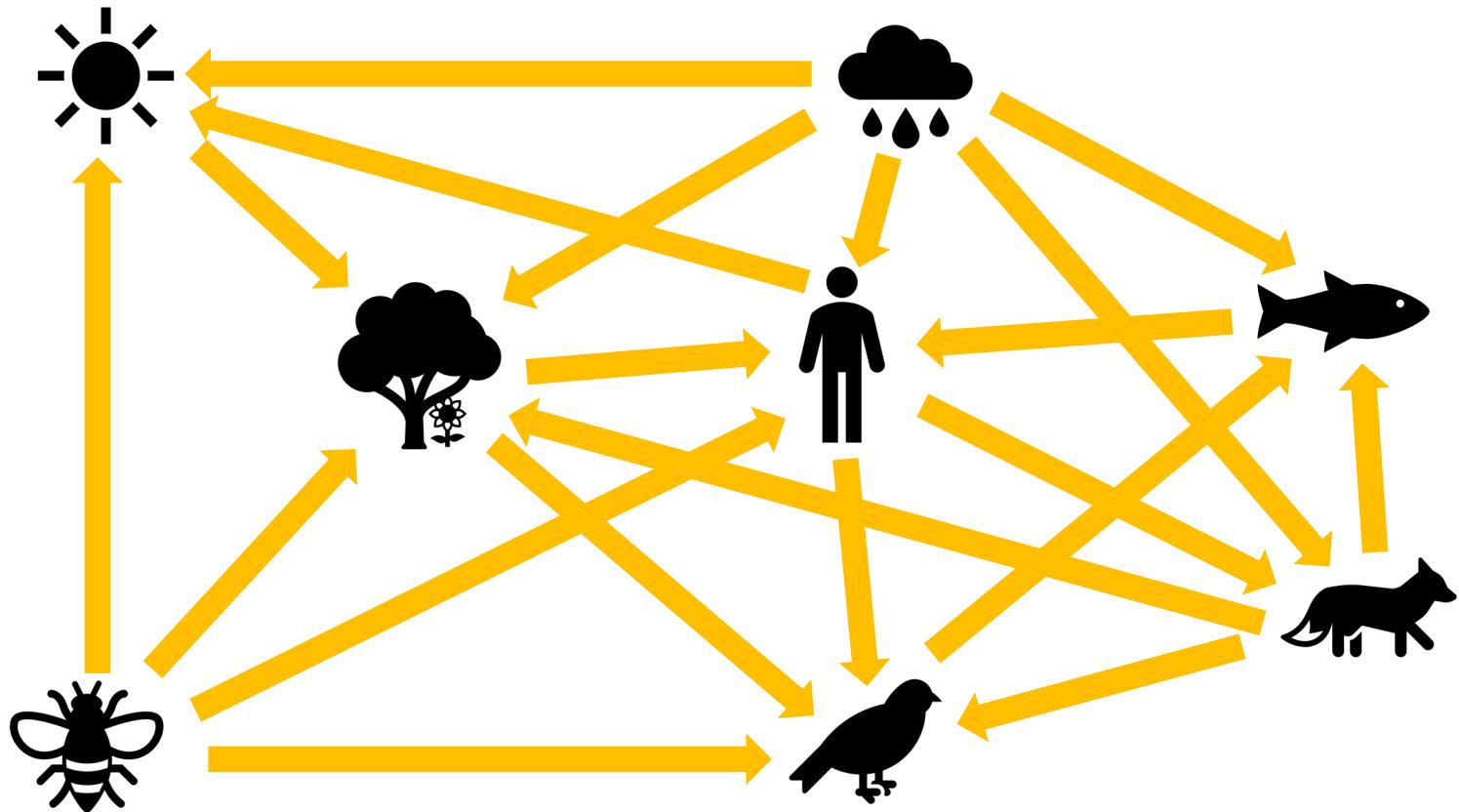
# How do we deal with causality?



Causal Directed Acyclic Graphs (DAGs)

# Intermezzo - causal discovery

obs.	☀	☁	🐟	👤	🌳	🐦
1	17°	1.8	19	100	42	12
2	22°	2.2	20	105	44	14
3	15°	0.9	21	106	39	8
4	18°	1.2	18	104	40	10
5	24°	0	17	100	41	11
6	19°	1.5	18	99	42	13
7	20°	1.6	18	103	58	7



# Intermezzo - causal discovery



# Intermezzo - causal discovery

CASE STUDY

GHENT UNIVERSITY & Undisclosed Swiss cosmetics researcher

Causal inference between metagenomics and odour

Data-driven causal discovery

SKILLS

ML/AI

Reporting

Genomics

Visualisation

Data mining

Workflow development

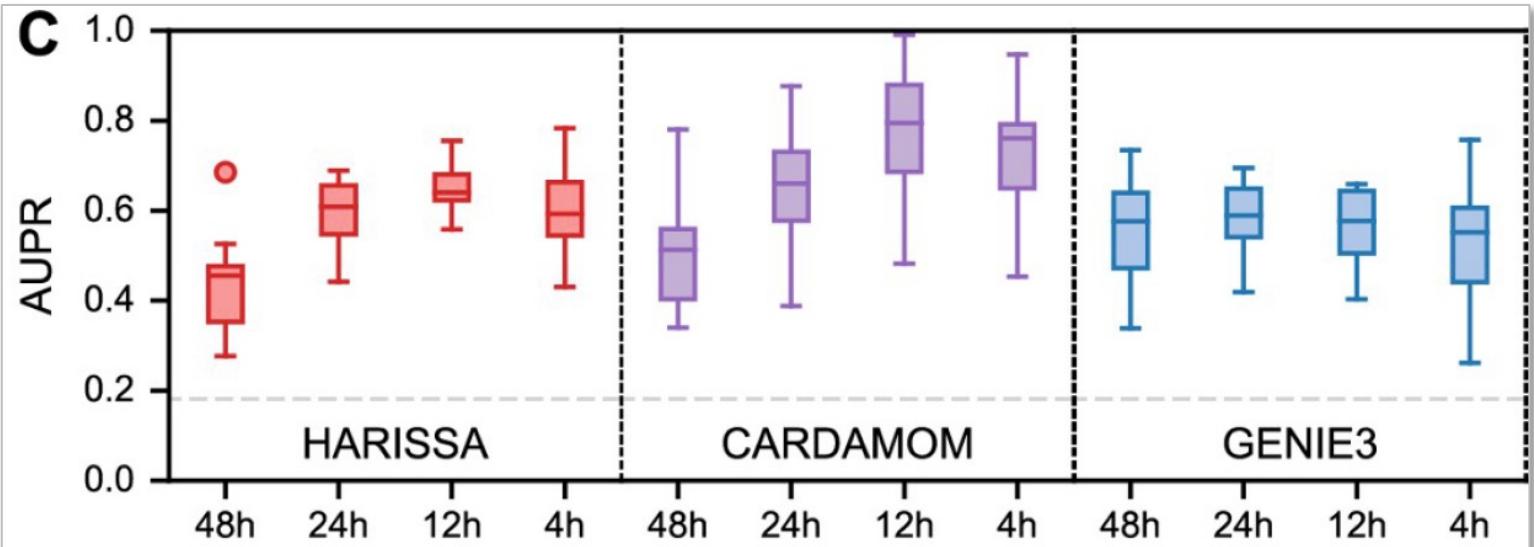
Biostatistics

BioLizard

The slide is a case study from Ghent University and an undisclosed Swiss cosmetics researcher. It focuses on causal inference between metagenomics and odour. A flow diagram shows data from a crowd being processed through a washing machine, a tailor, and a computer, resulting in 598 causal relations. A network graph on the right illustrates these relationships. The bottom section lists skills: ML/AI, Reporting, Genomics, Visualisation, Data mining, Workflow development, and Biostatistics, all associated with the BioLizard logo.

# Intermezzo - causal discovery

Interesting field, but state-of-the-art is still not very accurate



[Submitted on 29 Aug 2023]

## The CausalBench challenge: A machine learning contest for gene network inference from single-cell perturbation data

Mathieu Chevalley, Jacob Sackett-Sanders, Yusuf Roohani, Pascal Notin, Artemy Bakulin, Dariusz Brzezinski, Kaiwen Deng, Yuanfang Guan, Justin Hong, Michael Ibrahim, Wojciech Kotlowski, Marcin Kowiel, Panagiotis Misiakos, Achille Nazaret, Markus Püschel, Chris Wendler, Arash Mehrjou, Patrick Schwab

### Nonlinear causal discovery with additive noise models

Part of Advances in Neural Information Processing Systems 21 (NIPS 2008)

Bibtex    Metadata    Paper

### Authors

Patrik Hoyer, Dominik Janzing, Joris M.

### Conditional Distribution Variability Measures for Causality Detection

Josè A. R. Fonollosa

Chapter | First Online: 23 October 2019

658 Accesses | 1 Citations

### Learning Functional Causal Models with Generative Neural Networks

Olivier Goudet, Divyyan Kalainathan, Philippe Caillou, Isabelle Guyon, David Lopez-Paz & Michèle Sebag

Chapter | First Online: 30 November 2018

3776 Accesses | 12 Citations | 1 Altmetric

Article | Open access | Published: 26 October 2023

### Deep learning of causal structures in high dimensions under data limitations

Kai Lagemann, Christian Lagemann, Bernd Taschler & Sach Mukherjee

Nature Machine Intelligence 5, 1306–1316 (2023) | Cite this article

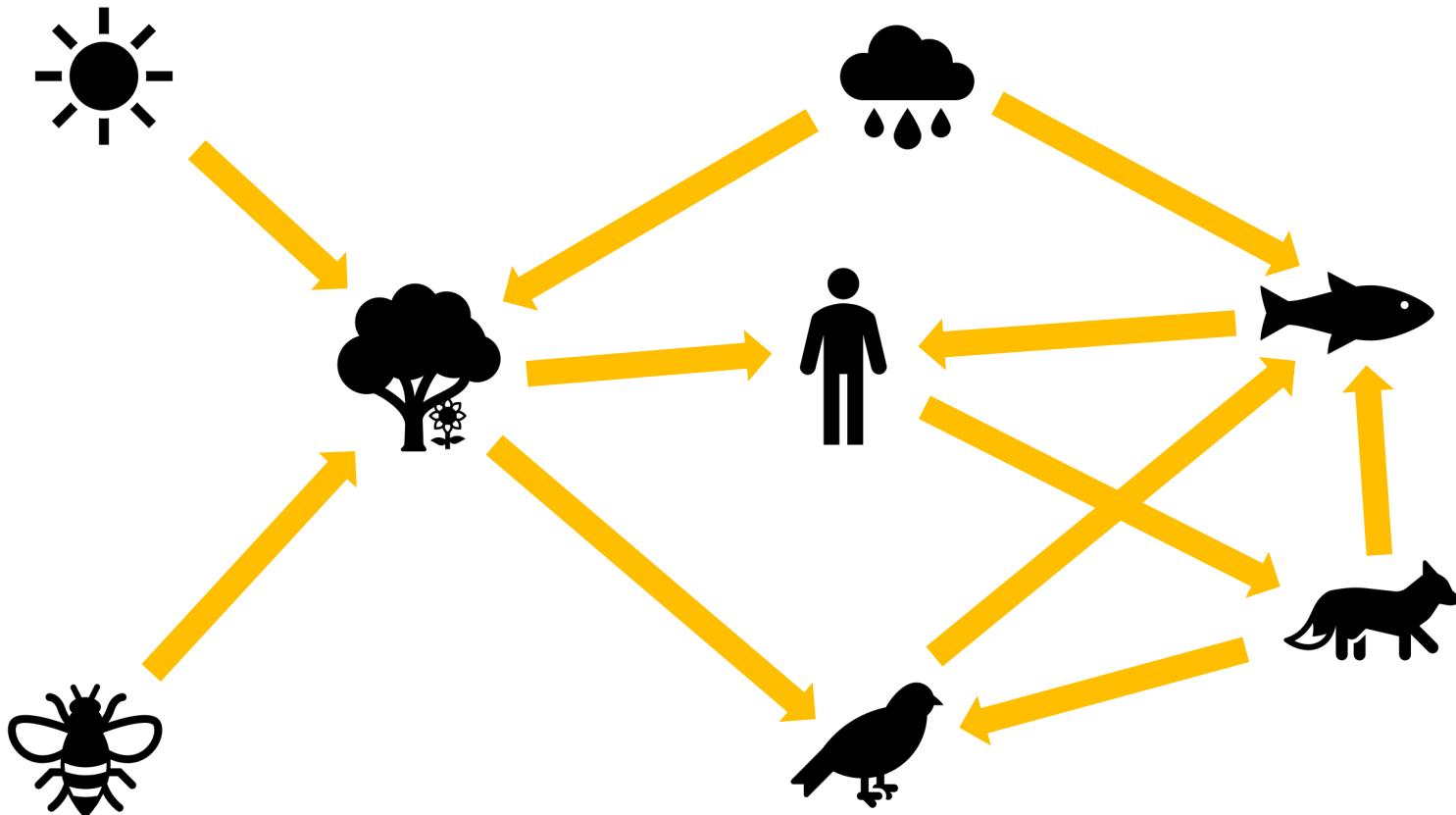
### Analysis of cause-effect inference by comparing regression errors

Research article | Artificial Intelligence | Data Mining and Machine Learning

Patrick Blöbaum<sup>1</sup>, Dominik Janzing<sup>2</sup>, Takashi Washio<sup>1</sup>, Shohei Shimizu<sup>3</sup>, Bernhard Schölkopf<sup>2</sup>

January 21, 2019

## Intermezzo - causal inference

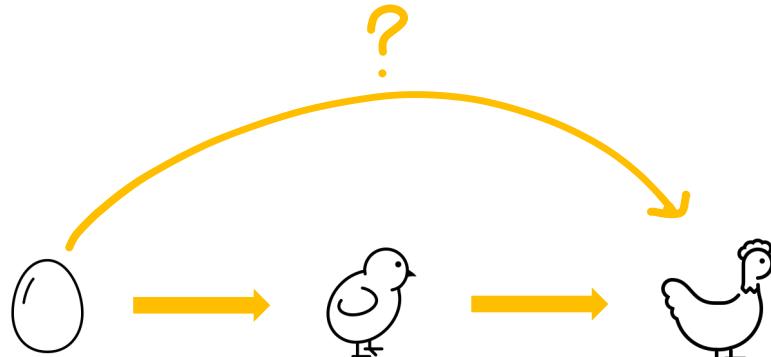


$$\text{cloud} \rightarrow \text{fish} : p = 0.007$$

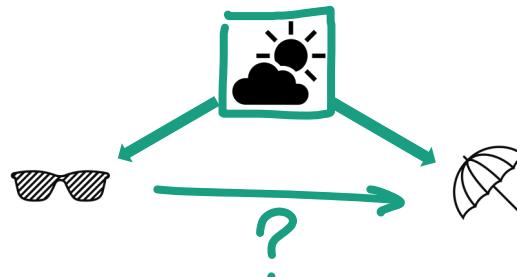
If  $\text{sun} \times 2$ , then  $\text{bird} \times 1.2$

Had  $\text{tree} \times 2$ , but  $\text{person} =$ ,  
then  $\text{fish} \times 0.8$

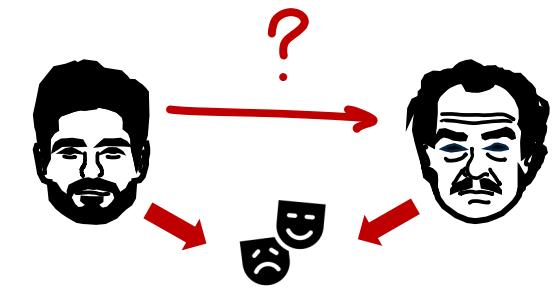
# How do we deal with causality?



Don't adjust!

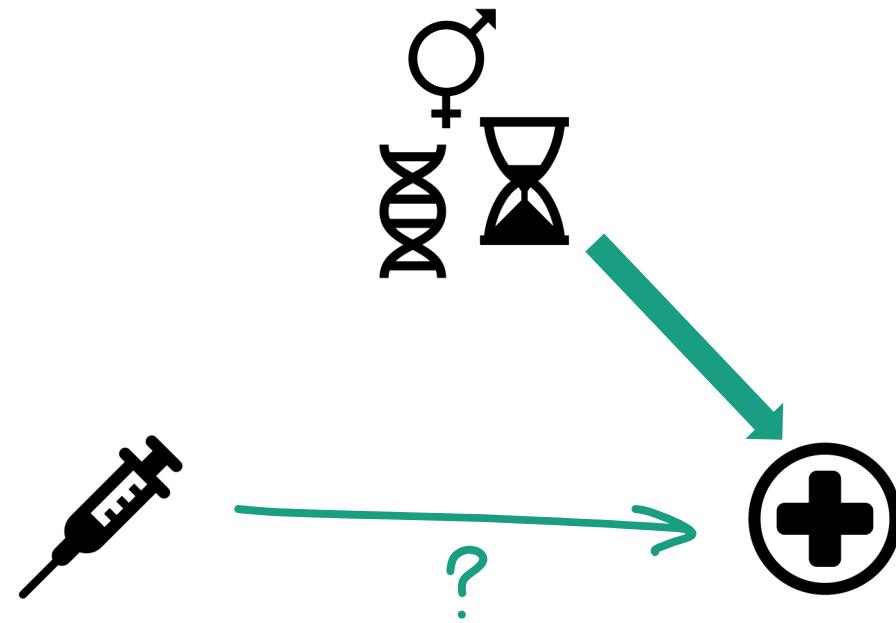


Adjust!



Don't adjust!

# Why do RCTs work?



# Causal inference



how.ipynb

```
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
import warnings
warnings.filterwarnings('ignore')
```

Part 2: How should we deal with causality?

### Low dimensional settings

When statisticians talk about bias, adjustment is often the go-to solution. When you have a **low dimensional dataset**, i.e., you only have a limited number of features, it is often possible to adjust for confounding variables. This is done by either stratifying (taking the average of the effects X on Y over each of the confounding groups) or by including the confounder variable in the model.

For example, if you are interested in the effect of a gene expression on a survival rate, as in Uhlen *et al.*, 2018, you might want to adjust for the age of the patient, as older patients are more likely to die sooner\*.

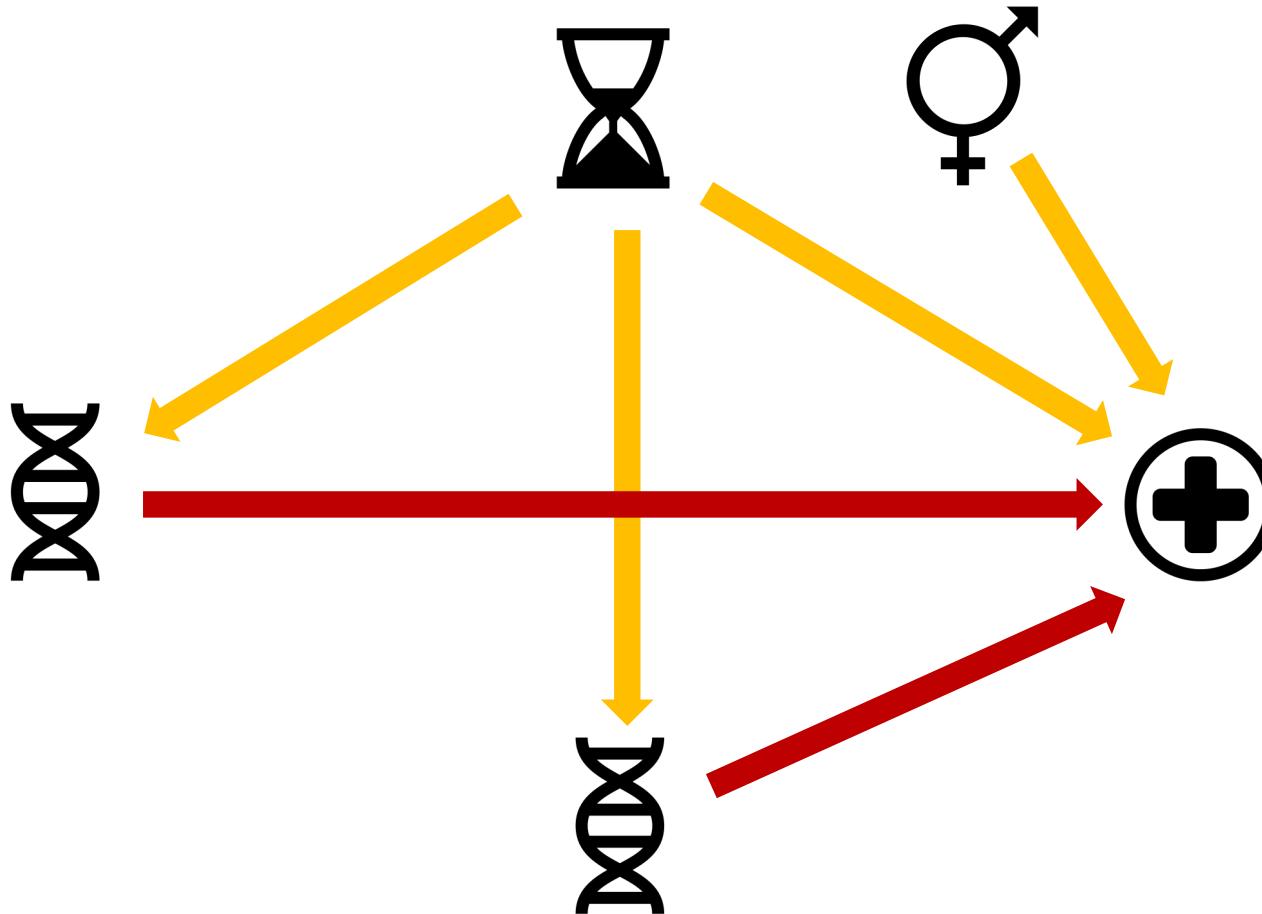
\* Interestingly, Uhlen *et al.* did not adjust for age in their 2018 Science publication. A highly interesting critical review of their methods can be found [here](#).

### Experiment 3: adjusting for confounding variables

The dataset in `data/transcriptomics_pathology.csv` contains metadata and gene expression data on 10 genes for 1000 patients:

```
data = pd.read_csv('data/transcriptomics_pathology.csv', index_col=0)
```

# Causal inference



A screenshot of a Jupyter Notebook interface. The left sidebar shows a file tree with files like 'whi.csv', 'whi\_synt', 'DAWNEY\_191', 'data', 'data/genes', 'data/transcriptomics', 'data/transcriptomics\_pathology', 'data/transcriptomics\_pathology.csv', 'index\_col=0', 'README.md', and 'whi\_synt'. The main area displays Python code and its output. The code reads a CSV file and prints a head of the data:

```
data = pd.read_csv('data/transcriptomics_pathology.csv', index_col=0)
data.head()
```

The output shows five patients with their age, gender, gene expression levels, and survival years:

	Age	Gender	Gene_1 expression	Gene_2 expression	Survival Years
Patient 1	Young	Woman	0.126698	0.853764	24
Patient 2	Old	Man	0.186956	0.203928	14
Patient 3	Old	Woman	0.343942	0.339771	65
Patient 4	Old	Man	0.096649	0.287072	16
Patient 5	Young	Man	0.198464	0.924111	60

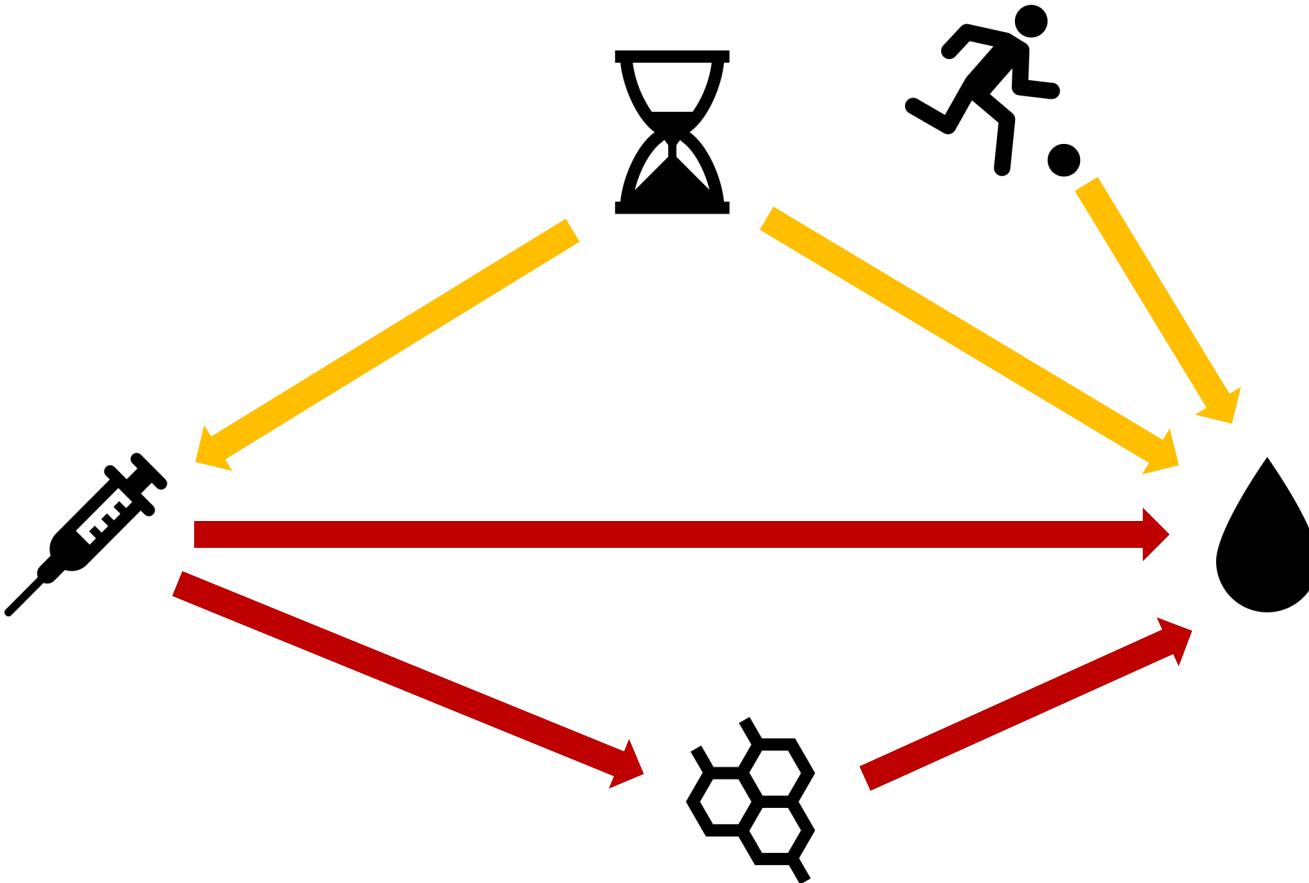
The code then performs linear regression to find causal relationships:

```
from statsmodels.formula.api import ols
data['Age'].replace('Young', '0', inplace=True)
data['Gender'].replace('Man', '0', inplace=True)
data['Survival Years'].replace('24', '1', inplace=True)
data['Survival Years'].replace('14', '0', inplace=True)
data['Survival Years'].replace('65', '1', inplace=True)
data['Survival Years'].replace('16', '0', inplace=True)

linear_model = ols('survival ~ Gene_1 + Gene_2 + Age', data).fit()
print(linear_model.summary())
```

The output indicates that Gene 1's expression has an impact on survival with coefficient 0.126698, and Gene 2's expression has an impact on survival with coefficient 0.853764.

# Causal inference



The screenshot shows a Jupyter Notebook interface with the following content:

```
data = pd.read_csv('~/data/blood_pressure.csv', index_col=4)
```

Experiment 4: All for adjustment, and adjustment for all

While adjustment is a step in the right direction, i.e., the one that acknowledges causality, it can sometimes do more harm than good. Consider the following example: Effects of a New Drug on Blood Pressure

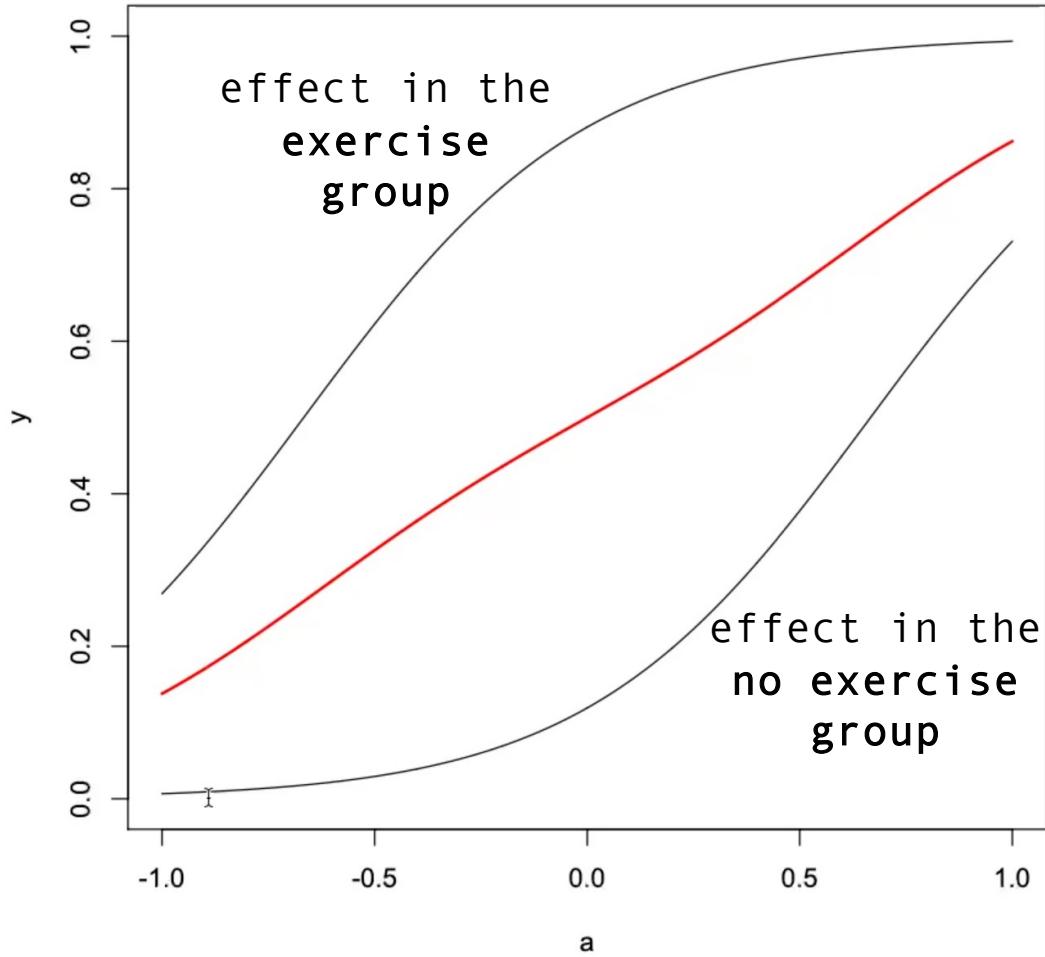
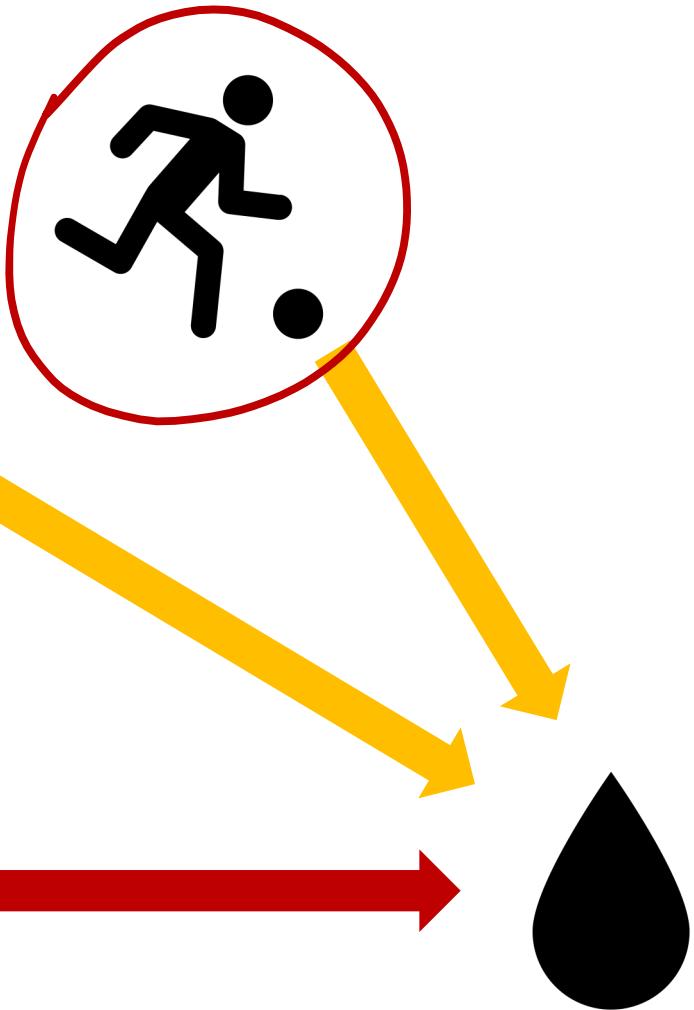
Imagine a clinical trial testing the efficacy of a new drug designed to lower blood pressure in hypertensive patients. The DAG for this scenario might look like this:

- Treatment: Represents whether a patient received the new drug (e.g., Drug X) or a placebo.
- Age: Represents the patient's age, a potential confounding variable associated with both the treatment and blood pressure.
- Exercise: Represents whether the patient regularly partakes in physical activity, which could affect blood pressure independently of the treatment.
- Outcome: Represents the measured level of the patient after receiving the treatment. This may impact blood pressure as well.

Which variables should you include in your model to estimate the causal effect of the treatment on blood pressure? How does including them influence the coefficient of the treatment variable?

	Treatment	Age	Exercise	Cholesterol	Blood Pressure
Patient 1	Placebo	Young	0	170.89025	169.727183
Patient 2	Treatment	Old	1	168.890741	124.321006
Patient 3	Placebo	Young	0	163.884529	113.691193
Patient 4	Placebo	Old	0	169.884529	120.321006
Patient 5	Placebo	Young	0	195.620208	119.844416

# Should you adjust for exercise?



Adjusted  
effect

You can but don't  
have to

The screenshot shows a Jupyter Notebook interface with the following content:

**Experiment 4: All for adjustment, and adjustment for all**

While adjustment is a step in the right direction, i.e., the one that acknowledges causality, it can sometimes do more harm than good. Consider the following example: Effects of a New Drug on Blood Pressure

Imagine a clinical trial testing the efficacy of a new drug designed to lower blood pressure in hypertensive patients. The DAG for this scenario might look like this:

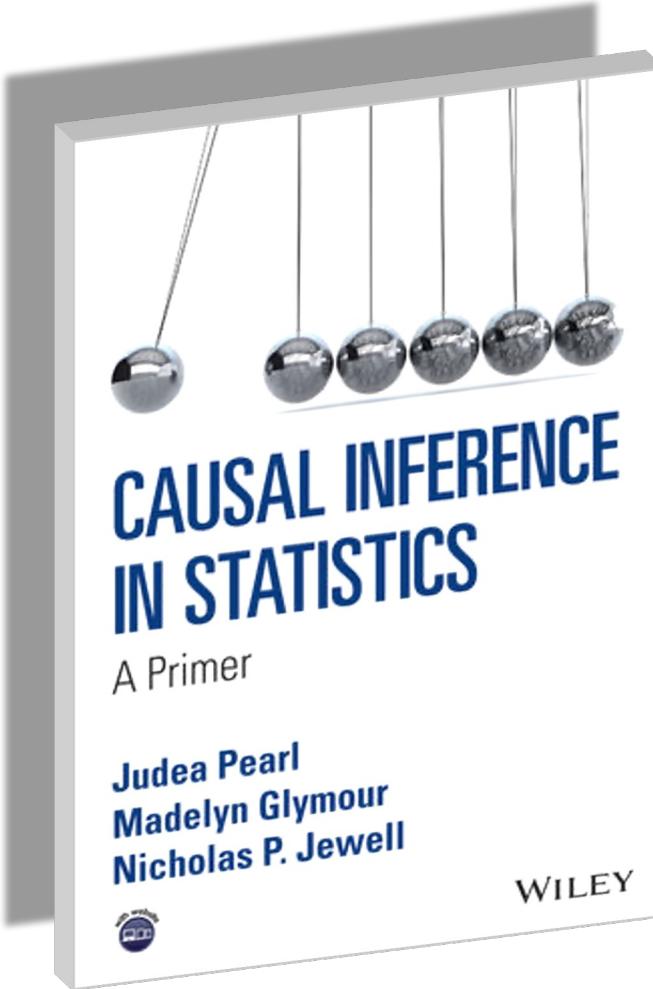
- Treatment: Represents whether a patient received the new drug (e.g., Drug X) or a placebo.
- Age: Represents the patient's age, a potential confounding variable associated with both the treatment and the outcome.
- Exercise: Represents whether the patient regularly partakes in physical activity, which could affect blood pressure independently of the treatment.
- Outcome: Represents the measured level of the patient after receiving the treatment. This may impact blood pressure as well.

data = pd.read\_csv('~/data/blood\_pressure.csv', index\_col=0)

Patient 1: Placebo Young 0 170.890758 109.773183  
Patient 2: Treatment Old 1 168.890741 124.321006  
Patient 3: Placebo Young 0 163.884459 113.691193  
Patient 4: Placebo Old 0 168.884459 113.691193  
Patient 5: Placebo Young 0 195.620208 115.844646

Which variables should you include in your model to estimate the causal effect of the treatment on blood pressure? How does including them influence the coefficient of the treatment variable?

# More about causal inference\*



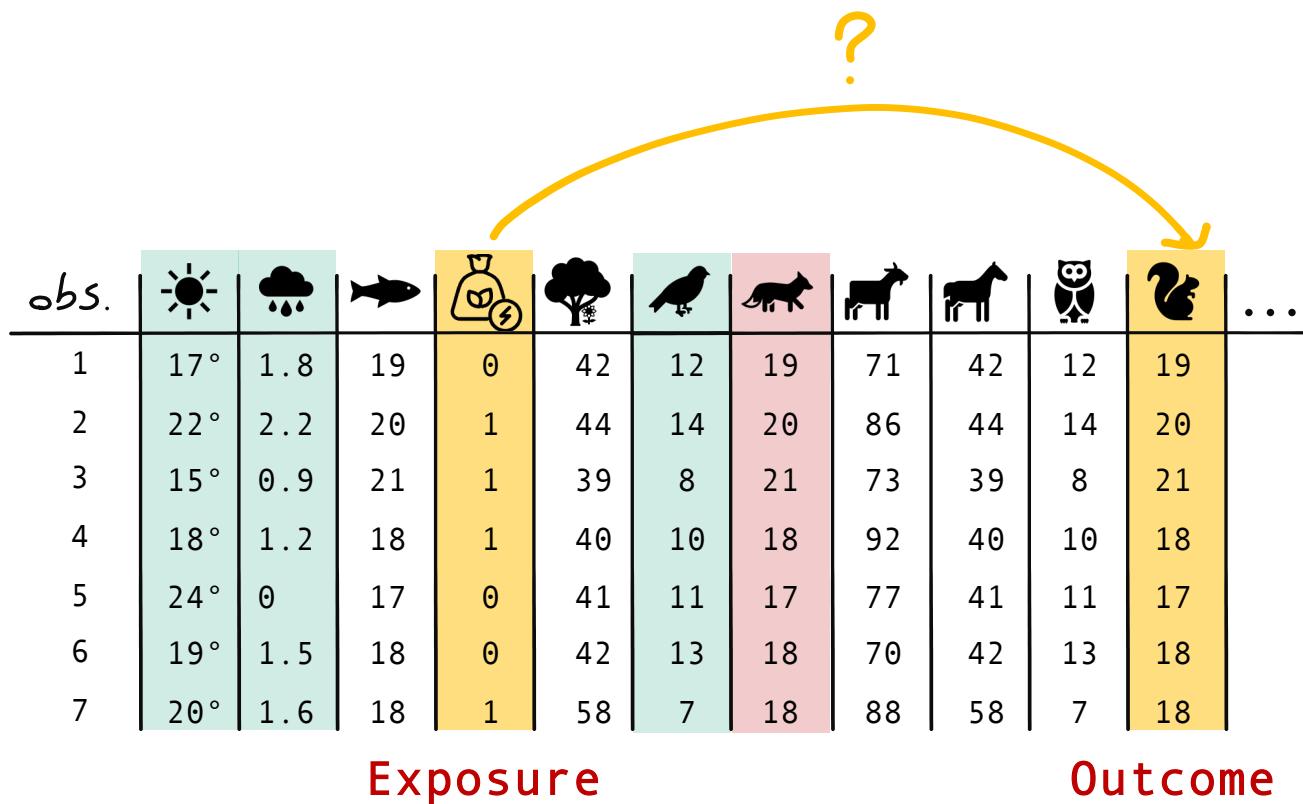
\*in low dimensional  
data settings

## What about high dimensional data?

obs.	sun	rain	fish	bag	tree	pigeon	wolf	goat	horse	owl	squirrel	dove	mouse	turtle	cat	chicken	skunk	...
1	17°	1.8	19	0	42	12	19	71	42	12	19	100	42	12	19	100	42	
2	22°	2.2	20	1	44	14	20	86	44	14	20	105	44	14	20	105	44	
3	15°	0.9	21	1	39	8	21	73	39	8	21	106	39	8	21	106	39	
4	18°	1.2	18	1	40	10	18	92	40	10	18	104	40	10	18	104	40	
5	24°	0	17	0	41	11	17	77	41	11	17	100	41	11	17	100	41	
6	19°	1.5	18	0	42	13	18	70	42	13	18	99	42	13	18	99	42	
7	20°	1.6	18	1	58	7	18	88	58	7	18	103	58	7	18	103	58	

**Exposure**      **Outcome**

# Causal machine learning



$$E[\hat{\mathbb{S}}^1 - \hat{\mathbb{S}}^0] = \frac{\sum_{i=1}^n \mathbb{S}_i (\hat{\mathbb{S}}_i - Q^{(0)}(L_i))}{\sum_{i=1}^n \mathbb{S}_i} - \sum_{i=1}^n (1 - \mathbb{S}_i) \frac{g(L_i)}{1 - g(L_i)} (\hat{\mathbb{S}}_i - Q^{(0)}(L_i))$$

Propensity model:

$$\hat{E}(\mathbb{S}_i = 1 | \text{☀️, ☁️, 🐦, ...}) \rightarrow g(L)$$

Outcome model:

$$\hat{E}(\hat{\mathbb{S}}_i | \mathbb{S}_i = 0, \text{☀️, ☁️, 🐦, ...}) \rightarrow Q^{(0)}(L)$$

# Causal machine learning



how.ipynb

The screenshot shows a Jupyter Notebook interface with several tabs open: 'why.ipynb', 'how.ipynb' (the active tab), 'CML\_Lab6\_Solution.R', and 'question\_3\_4\_Taelman.Rmd'. The 'how.ipynb' tab displays a section titled 'High dimensional settings' which discusses the challenges of estimating causal effects in high-dimensional datasets. It mentions the use of the Augmented Inverse Probability Weighted estimator (AIPW). Below this, another section titled 'Experiment 5: causal machine learning' is shown, along with a snippet of Python code for reading a CSV file:

```
data = pd.read_csv('./data/lindner.csv', index_col=0)
data.head()
```

The interface also includes a terminal window showing command-line activity, a problems panel with 284 errors, and various toolbars and status bars at the bottom.

# Causal machine learning - example



Interventional Cardiology

## Abciximab provides cost-effective survival advantage in high-volume interventional practice

Dean J. Kereiakes MD<sup>a</sup>, Robert L. Obenchain PhD<sup>b</sup>, Beth L. Barber PhD<sup>b</sup>, Andrew Smith BS<sup>a</sup>,  
Mark McDonald MA<sup>c</sup>, Thomas M. Broderick MD<sup>d</sup>, John Paul Runyon MD<sup>d</sup>,  
Thomas M. Shimshak MD<sup>d</sup>, John F. Schneider MD<sup>d</sup>, Charles R. Hattemer MD<sup>d</sup>, Eli M. Roth MD<sup>d</sup>,  
David D. Whang MD<sup>d</sup>, Douglas Cocks PhD<sup>b</sup>, Charles W. Abbottsmith MD<sup>d</sup>

The screenshot shows a Jupyter Notebook interface with the title "Experiment 5: causal machine learning". The code cell contains Python code to read a CSV file and print its head:

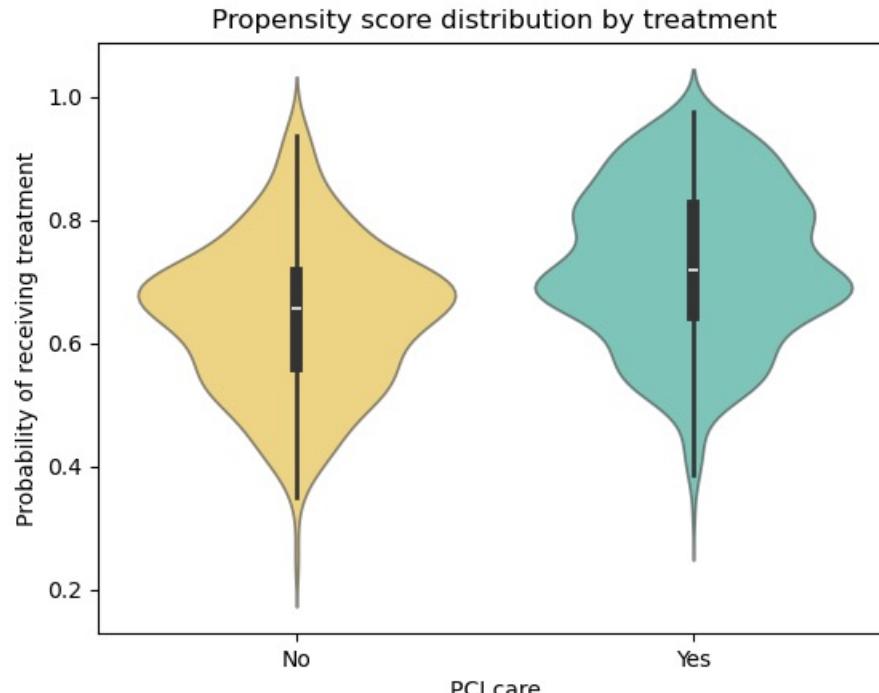
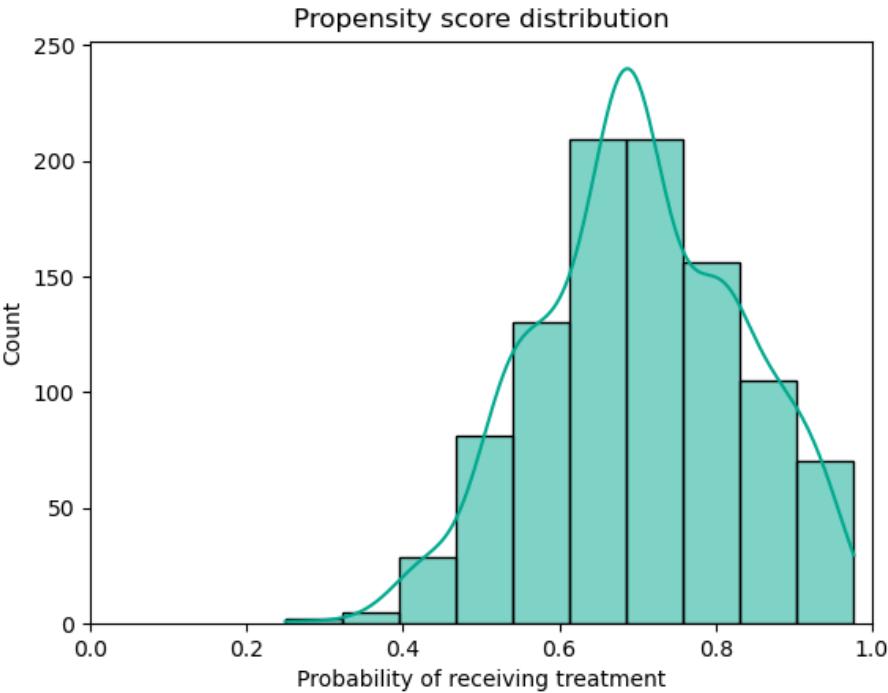
```
data = pd.read_csv('./data/linder.csv', index_col=0)
data.head()
```

The output cell displays the first few rows of the "linder" dataset:

	PCI care	Stent	Height	Female	Diabetic	Myocardial infarction	Ejection fraction	Vessels	Died
1	1	1	163	1	1	0	56	1	1
2	1	1	163	0	0	0	55	1	0
3	1	0	168	0	0	0	50	1	0
4	1	0	175	0	1	0	50	1	0
5	0	1	168	1	0	0	55	1	0

A note at the bottom of the notebook states: "This is not a randomized controlled study (not a randomized controlled trial), patients received the treatment based on the physician's decision, not the MD. We first build the propensity model, which estimates the probability of receiving the treatment given the observed covariates."

# Causal machine learning - example

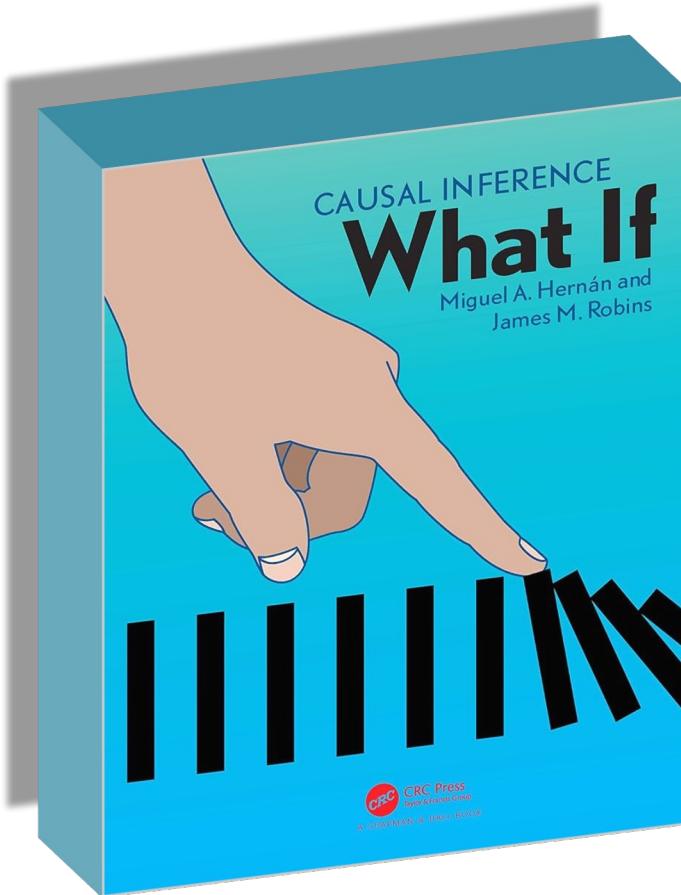


The screenshot shows a Jupyter Notebook interface with several tabs open. The main code cell contains:

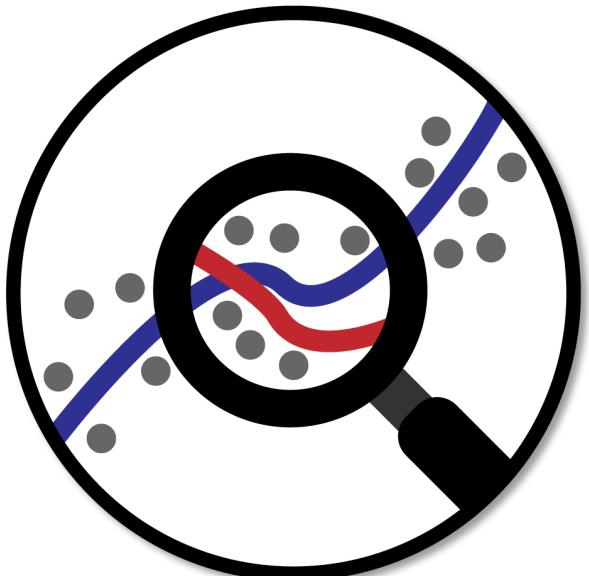
```
data = pd.read_csv('data/linnerer.csv', index_col=0)
```

Below the code, a note states: "For this example, we will work on the Linner dataset, an observational study of 996 patients receiving an initial Percutaneous Coronary Intervention (PCI) at Ohio Heart Hospital, Christ Hospital, Cincinnati in 1997 and followed for at least 6 months by the end of the study (source: Berkenblit et al., 2000)." The notebook also lists other files like README.md, whyalgotest.ipynb, and environment.yml.

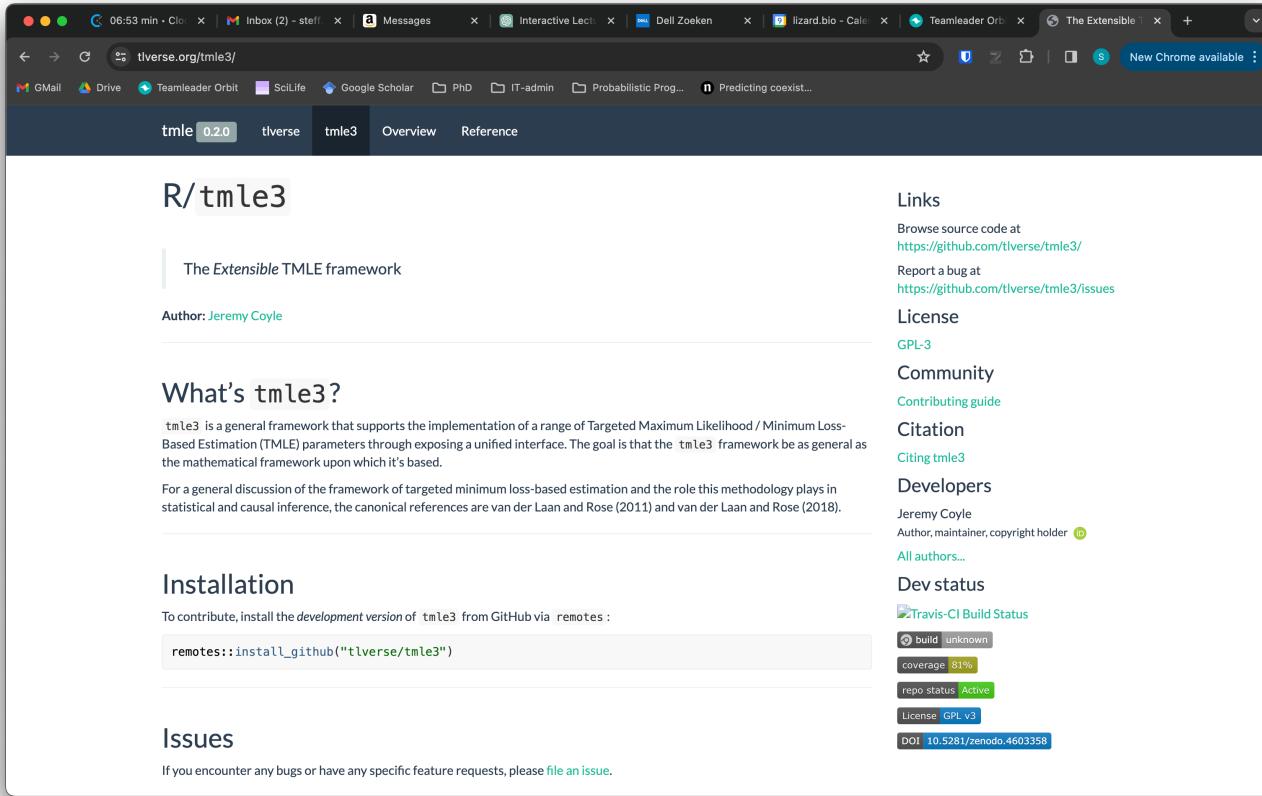
# More about causal machine learning



# More about causal machine learning



tlverse



R/tmle3

The Extensible TMLE framework

Author: Jeremy Coyle

What's tmle3?

tmle3 is a general framework that supports the implementation of a range of Targeted Maximum Likelihood / Minimum Loss-Based Estimation (TMLE) parameters through exposing a unified interface. The goal is that the tmle3 framework be as general as the mathematical framework upon which it's based.

For a general discussion of the framework of targeted minimum loss-based estimation and the role this methodology plays in statistical and causal inference, the canonical references are van der Laan and Rose (2011) and van der Laan and Rose (2018).

Installation

To contribute, install the development version of tmle3 from GitHub via remotes :

```
remotes::install_github("tlverse/tmle3")
```

Issues

If you encounter any bugs or have any specific feature requests, please [file an issue](#).

Links

- Browse source code at <https://github.com/tlverse/tmle3/>
- Report a bug at <https://github.com/tlverse/tmle3/issues>

License

GPL-3

Community

Contributing guide

Citation

Citing tmle3

Developers

Jeremy Coyle

Author, maintainer, copyright holder

All authors...

Dev status

Travis-CI Build Status

- build unknown
- coverage 81%
- repo status Active
- License GPL v3
- DOI 10.5281/zenodo.4603358

# Causality 101

## How to draw causal statements from data and get away with it

Lizard Lab - 09.02.2024

# Causality 101

Lizard Lab - 09.02.2024